5

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/612,802	07/02/2003	John M. Lake	RSW920030082US1	5358

48816 7590 06/04/2007

IBM CORPORATION - RSW (JVL)
C/O VAN LEEUWEN & VAN LEEUWEN
P.O. BOX 90609
AUSTIN, TX 78709-0609

EXAMINER	
ROSE, HELENE ROBERTA	
ART UNIT	**PAPER NUMBER**
2163	

MAIL DATE	DELIVERY MODE
06/04/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/612,802	LAKE, JOHN M.	
	Examiner	Art Unit	
	Helene Rose	2163	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 March 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-26* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-26* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *02 July 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Detailed Action

1. In response to communication filed on 3/13/2007, Claims 1, 3, 7-10, 12, 14-17, and 21-26 have been amended. No claims were cancelled nor added. Therefore, Claims 1-26 is pending.

2. Applicant's arguments filed with respect to the rejected claims have been fully considered but they are not persuasive.

Claim Rejections – 35 U.S.C 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-26 are rejected under 35 U.S.C. 103(a) as being obvious over Sayag (US Patent No, 6,898,602/Filing Date of Patent: December 10, 2002) in view of Kundu et al (US Publication No. 20040210577/Filing Date of Publication: April 16, 2003) and further in view of Kolawa et al (US Patent No. 5,842,019, Date of Patent: November 24, 1998)

Claims 1,10,15,and 24-26:

Regarding claims 1,10,15, and 24-26, although claims 1 and 24 teach a method, claims 15 and 26 teach a computer program product and claims 10 and 25 teach an

information handling system. Thus, the following claims 1,10,15, and 24-26 implements the same limitations to carry out the invention.

Sayag teaches a method/computer program product/system for automatically nullifying (column 2, lines 23-29) variables in a middleware computer program (see Figure 1, all features, Sayag), said method/computer program product/system comprising:

one or more processors (column 4, lines 53-57 and column 2, lines 64-65, wherein a data processing system is known as a system that includes computer systems and associated personnel, that performs input, processing storage, output, and control functions to accomplish a sequence of operations on data, Sayag);

a memory accessible by the processors (columns 4-5, lines 65-67 and lines 1-5, Sayag);

a middleware software application that runs on the operating system (column 5, lines 8-10, wherein a middleware software application is known as a communication layer that allows applications to interact across hardware and network environments, Sayag), the middleware application including a garbage collected heap (column 2, lines 29-30, wherein the garbage collector is invoked at each instruction, Sayag); and

a nullification tool for nullifying program references (column 2, lines 64-67, Sayag[1]), the nullification tool comprising steps effective to:

[1] The Examiner interprets the term "nullification tool" to be an act of nullifying; making null and void; counteracting or overriding the effect or force of something. Therefore the tool utilized within Sayag invention that carries out the same function of a nullification tool is identified within (columns 6-7, lines 66-67 and lines 1-17 and column 8, line 63, Sayag).

reading one or more variables included in one or more activation records included in the **middleware software application** (column 5, lines 17-38, wherein reading variable excessive_gc is made and reading variable trace_usage is made, and if variable is set then a display if memory usage is activated and column 6, lines 1-5, wherein one activation record mallocHeapObject is disclosed and a program statement is read from working memory and evaluated, Sayag);

Sayag discloses all the limitations above. However, Sayag does not discloses wherein identifying a program statement where the variable is last used nor does he disclose inserting a nullification statement after the identified program,

On the other hand, Kundu discloses identifying a program statement in the **middleware software application** where **a selected variable** is last used (paragraph [0051], wherein the remaining field contain information about the current state of the session, wherein locations in physical redo logs are identified by system change numbers, or SCN's. start_scn and end_scn are the SCN's of the physical redo blocks at which the logical redo log made for the session will begin and end and checkpoint_scn is the SCN of the most recently-made checkpoint in the logical redo log, which is equivalent to identifying a program statement in the program where the variable is last used, and wherein a checkpoint is interpreted to be an identified snapshot of a database or a point at which the transactions against the database have been frozen, Kundu); and Kundu teaches ~ inserting a nullification statement after the identified program (paragraph [0066], wherein wherein the builder also makes checkpoints in logical redo log by inserting

checkpoint logical change records, i.e. LCR at the proper locations in logical redo log, and wherein a redo log is interpreted to be a set of files that record all changes made to an database and a checkpoint is interpreted to be an identified snapshot of a database or a point at which the transactions against the database have been frozen, which is equivalent to nullifying the identified last used variable, Kundu).

It would have been obvious to one of the ordinary skill in the art at the time of the invention was made to incorporate Kundu teaching into Sayag system to recognize when a variable was last used within a program. A skilled artisan would have been motivated to do so for allocating more space within memory to store data as well as maintaining memory management in a system.

Sayag in view of Kundu discloses the limitation above. However, Sayag in view of Kundu does not disclose the step of, wherein, the nullification statement adapted to nullify **the selected** variable.

On the other hand, Kolawa disclose the nullification statement adapted to nullify the identified last-used variable (column 5, lines 40-61, wherein track using allocated memory space using reference counting, if value is stored in a pointer and so forth, and wherein if the reference count for the pointer variable becomes zero, wherein zero is equivalent to nullify, Kolawa);

It would have been obvious to one of the ordinary skill in the art at the time of the invention was made to incorporate Sayag in view of Kundu teachings into Kolawa system. A skilled artisan would have been motivated to combine as suggested by Kolawa

at (column 1, lines 60-62 and column 2, lines 6-9, Kolawa}, for providing a code to identify and track memory space. As a result, for detecting leak memory space in a computer program.

writing a plurality of program statements (column 6, lines 29-32, wherein a determination is made whether the program statements remain to be written, if the decision step is positive, Sayag), including the identified program statement, to a resulting code file (column 6, lines 20-23, Sayag); and writing the nullification statement to the resulting code file (column 6, lines 20-23, Sayag) in a position subsequent to the identified program statement (column 3, lines 11-14, wherein after running the garbage collector, and determining the amount of the memory that is still in use of the heap, Sayag).

Claims 2,11, and 16:

Regarding claims 2,11, and 16, the combination of Sayag in view of Kundu and further in view of Kolawa teaches wherein the means for reading, means for identifying (column 3, lines 15-19, Sayag), and means for inserting are each performed by a compiler (column 5, lines 47-52, wherein a compiler is any program that transfer one set of symbols into another by a set of semantic rules, Sayag).

Claims 3,12,and 17:

Regarding claims 3,12, and 17, the combination of Sayag in view of Kundu and further in view of Kolawa teaches the computer program product further comprising:

means for writing the activation records, program statement (column 6, lines 29-32, wherein a determination is made whether the program statements remain to be written, if the decision step is positive, Sayag), and nullification statement to a resulting code file (column 6, lines 20-23, Sayag).

Claims 4,13, and 18:

Regarding claims 4,13, and 18, the combination of Sayag in view of Kundu and further in view of Kolawa teaches wherein at least one of the variables reference an object stored in a garbage collected memory heap (column 3, lines 1-14, Sayag).

Claims 5 and 19:

Regarding claims 5 and 19, the combination of Sayag in view of Kundu and further in view of Kolawa teaches wherein the activation records include one or more local variable definitions (column 6, lines 6-9, Sayag).

Claims 6 and 20:

Regarding claims 6 and 20, the combination of Sayag in view of Kundu and further in view of Kolawa teaches wherein the activation records include one or more argument parameters (column 5, lines 21-22, wherein parameter is defined, Sayag).

Claims 7,14, and 21:

Regarding claims 7,14, and 21, the combination of Sayag in view of Kundu and further in view of Kolawa teaches wherein the objects are stored in a garbage collected heap stored in a computer memory (column 3, lines 1-14), the method further comprising:

means for executing a garbage collection program (column 2, lines 29-30, wherein the garbage collector is invoked at each instruction, Sayag);

means for identifying (column 3, lines 7-8, Sayag), by the garbage collection program (column 2, lines 29-30, Sayag), one of the objects that was previously referenced by one of the variables included in the nullification statement (column 2, lines 42-44, Sayag); and

means for reclaiming the memory occupied by the identified object (column 2, lines 44-48, Sayag).

Claims 8 and 22:

Regarding claims 8 and 22, the combination of Sayag in view of Kundu and further in view of Kolawa teaches the computer program product further comprising:

means for executing a compiler to perform the reading (column 3, lines 6-8, wherein executing code of the application and column 5, lines 47-52, wherein a compiler is any program that transfer one set of symbols into another by a set of semantic rules, Sagay);

identifying (column 3, lines 7-8, Sayag) and inserting (column 5, lines 48-52, wherein the application may be installed in a memory as software and column 7, lines 37-38, wherein installed in a development program, Sayag);

means for writing a plurality of program statements including the program statement (column 6, lines 29-32, wherein a determination is made whether the program

statements remain to be written, if the decision step is positive, Sayag) to a resulting code file (column 6, lines 20-23, Sayag);

means for writing the nullification statement to the resulting code file (column 6, lines 20-23, Sagay) in a position subsequent to the identified program statement (column 3, lines 11-14, wherein after running the garbage collector, and determining the amount of the memory that is still in use of the heap, Sayag).

Claims 9 and 23:

Regarding claims 9 and 23, the combination of Sayag in view of Kundu and further in view of Kolawa teaches the computer program product further comprising:

means for identifying one or more statements from the plurality of statements (column 3,lines 15-19, Sayag) where one or more other objects are last used (column 7, line 48, wherein last instruction of the application is identified, Sayag); and

means for writing nullification statements (column 6, lines 29-32, Sayag) for each of the other objects following the identified statement corresponding to the object's last use to the resulting code file (column 3, lines 11-19, Sayag).

Examiner's Response to Applicant's Arguments

Applicant States:

Status of the Claims

Claims 1-26 are currently present in the Application, and claims 1, 10, 15, 24, 25, and 26 are independent claims. Claims 1, 3, 7-10, 12, 14-17, and 21-26 have been amended to correct potential antecedent basis issues with these claims. In addition, independent claims 15 and 26 have been amended to prevent any future issues with regard to 35 U.S.C. § 101. In particular, the preambles of independent claims 15 and 26 have been amended to clarify that the computer readable media includes instructions for execution by a computer which, when executed by the computer, cause the computer to perform the claimed method. Support for the amendments to the computer program product claims is found, for example, in Applicant's specification on page 16, lines 1-18. Claims 16, 17, 21, 22, and 23 have been amended to correspond to the amendments in independent claim 15. No claims have been canceled or added in this Response, and no new matter has been added as a result of the amendments.

Applicant Argues:

Claim Rejections -Alleged Obviousness Under 35 U.S.C.~ 103

Claims 1-26 stand rejected under 35 U.S.C. § 103(a) as being obvious over Sayag, U.S. Patent No. 6,898,602 (hereinafter Sayag), in view of Kundu et al., U.S. Publication No. 2004/0210577 (hereinafter Kundu), and further in view of Kolawa et al., U.S. Patent No.

5,842,019 (hereinafter Kolawa). Applicant respectfully traverses the rejections under 35 U.S.C. § 103.

A. There Is No Motivation To Combine Saya.q, Kundu, And Kolawa MPEP § 706.02(j) states, inter alia:To establish a prima facie case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings.

FACT THAT REFERENCES CAN BE COMBINED OR MODIFIED IS NOT SUFFICIENT TO ESTABLISH PRIMA FACIE OBVIOUSNESS

The mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also su.q.qests the desirability of the combination. In re Mills, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990) (Claims were directed to an apparatus for producing an aerated cementitious composition by drawing air into the cementitious composition by driving the output pump at a capacity greater than the feed rate. The prior art reference taught that the feed means can be run at a variable speed, however the court found that this does not require that the output pump be run at the claimed speed so that air is drawn into the mixing chamber and is entrained in the ingredients during operation. Although a prior art device "may be capable of being modified to run the way the apparatus is claimed, there must be a suqqestion or motivation in the reference to do so." 916 F.2d at 682, 16

USPQ2d at 1432.). See also In re Fritch, 972 F.2d 1260, 23 USPQ2d 1780 (Fed. Cir. 1992) (flexible landscape edging device which is conformable to a ground surface of varying slope not suggested by combination of prior art references).

Regarding the three references cited in the Office Action, Sayag discloses the "use of garbage collection in order to determine the exact amount of memory that is consumed by a running application at any point of its execution" (see Sayag, Abstract). Kolawa discloses "dynamically detecting leaked memory space in a computer program" (see Kolawa, Abstract). Both Sayag and Kolawa deal with the general technology area of runtime memory space. Sayag is concerned with determining the amount of memory space used by a runtime application, while Kolawa is concerned with detecting memory space leaks during runtime. Kundu, on the other hand, is concerned with a completely different technology area. Kundu discloses "techniques for making light-weight checkpoints in logs of streams of transactions" (see Kundu, Abstract). Kundu uses the redo logs that are typically used in database systems to log transactions, and purports to increase the usefulness of these redo logs for purposes such as data mining and replication of transactions. As discussed in detail in Kundu, a logical redo log is made from a physical redo log. Light-weight checkpoints are made and used in the logical redo log, and these light-weight checkpoints are used in data mining and replication (Kundu, paragraph [0041]). Kundu is not concerned with, and indeed does not even address, the issue of memory space as used by runtime applications. Kundu is in a completely different technology area from Sayag and Kolawa. Kundu is concerned with database and

DBMS technology, whereas Sayag and Kolawa deal with runtime memory space. Other than the fact that all three references have to do with the very general area of computer science, there is simply no justification to combine Kundu with either Sayag or Kolawa.

It appears that the Office Action improperly uses Applicant's claims as "guideposts" in selecting the references and simply concludes that it is "obvious" to combine the references. In doing so, Applicant asserts that the Office Action uses impermissible hindsight in combining Kundu with Sayag and Kolawa in order to support a rejection of Applicant's claims. As an example, in the rejection of claim 1, the Office Action cites Sayag as disclosing Applicant's first element ("reading one or more variables . . ."). The Office Action then cites Kundu as disclosing Applicant's second element ("identifying a program statement . . .") and the first part of Applicant's third element ("inserting a nullification statement..."). The Office Action then cites Kolawa as disclosing the remainder of Applicant's third element ("the nullification statement adapted to nullify . . ."). As discussed in further detail below, the Office Action cites Kundu as disclosing the step of "inserting a nullification statement" and then cites Kolawa as disclosing the definition of the nullification statement, i.e. "the nullification statement adapted to nullify the selected variable." Dissecting Applicant's claims in this manner and then using hindsight to pick and choose a variety of references to read on selected sections of Applicant's claim elements is not permissible. As stated in MPEP § 2143.03, "the mere fact that references can be combined or modified does not render the

resultant combination obvious unless the prior art also suggests the desirability of the combination" (emphasis added).

In this case, the prior art simply does not suggest the desirability of combining these references. Applicant submits that the Office Action fails to satisfy the burden set forth in MPEP §§ 706.02(j) and 2143.03 in support of an obviousness objection, particularly because there is no motivation to combine the references. Thus, Applicant contends that the Office Action uses impermissible hindsight in rejecting Applicant's claims. For the reasons set forth above, Applicant respectfully submits that claims 1-26 are not obvious, and are therefore patentable over Sayag in view of Kundu and Kolawa.

Examiner's Response:

(1) In response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988) and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, Kundu teaches "identifying a program statement in the program where the variable is last used" paragraph [0051], wherein the remaining field contain information about the current state of the session, wherein locations in physical redo logs are identified by system change

numbers, or SCN's. start_scn and end_scn are the SCN's of the physical redo blocks at which the logical redo log made for the session will begin and end and checkpoint_scn is the SCN of the most recently-made checkpoint in the logical redo log, which is equivalent to identifying a program statement in the program where the variable is last used, and wherein a checkpoint is interpreted to be an identified snapshot of a database or a point at which the transactions against the database have been frozen, Kundu); and Kundo teaches "inserting a nullification statement after the identified program" (paragraph [0066], wherein the builder also makes checkpoints in logical redo log by inserting checkpoint logical change records, i.e. LCR at the proper locations in logical redo log, and wherein a redo log is interpreted to be a set of files that record all changes made to an database and a checkpoint is interpreted to be an identified snapshot of a database or a point at which the transactions against the database have been frozen, which is equivalent to nullifying the identified last used variable, Kundu). Therefore, It would have been obvious to one of the ordinary skill in the art at the time of the invention was made to incorporate Kundu {see abstract} teaching into Sayag system to recognize when a variable was last used within a program. A skilled artisan would have been motivated to do so for allocating more space within memory to store data as well as maintaining memory management in a system.

AND wherein: Sayag in view of Kundu do not teach, "the nullification statement adapted to nullify the identified last-used variable".

On the other hand, Kolawa disclose the nullification statement adapted to nullify the identified last-used variable (column 5, lines 40-61, wherein track using allocated memory space using reference counting, if value is stored in a pointer and so forth, and wherein if the reference count for the pointer variable becomes zero, wherein zero is equivalent to nullify, Kolawa). Therefore, It would have been obvious to one of the ordinary skill in the art at the time of the invention was made to incorporate Sayag in view of Kundu teachings into Kolawa system. A skilled artisan would have been motivated to combine as suggested by Kolawa at {column 1, lines 60-62 and column 2, lines 6-9, Kolawa}, for providing a code to identify and track memory space. As a result, for detecting leak memory space in a computer program.

(2) In response to applicant asserts that the Office Action use "impermissible hindsight" in combining prior art: Kundu, Sayag and Kolawa. Examiners states, that in response to applicant's argument that the examiner's conclusion of obviousness is based upon improper hindsight reasoning, it must be recognized that any judgment on obviousness is in a sense necessarily a reconstruction based upon hindsight reasoning. But so long as it takes into account only knowledge which was within the level of ordinary skill at the time the claimed invention was made, and does not include knowledge gleaned only from the applicant's disclosure, such a reconstruction is proper. See *In re McLaughlin*, 443 F.2d 1392, 170 USPQ 209 (CCPA 1971).

Applicant States/Argues:

B. The Cited References Do Not Teach Or Suqqest All The Limitations Of Applicant's Claims

To establish prima facie obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. In re Royka, 490 F.2d 981, 180 USPQ 580 (CCPA 1974). "All words in a claim must be considered in judging the patentability of that claim against the prior art." In re Wilson, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970). If an independent claim is nonobvious under 35 U.S.C. 103, then any claim depending therefrom is nonobviousness. In re Fine, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988). Manual of Patent Examining Procedure § 2143.03. Applicant respectfully submits that none of the cited references, either alone or in combination, teaches or suggests all the elements of Applicant's claims.

Using independent claim 1 as an exemplary claim, Applicant teaches and claims the following: reading one or more variables included in one or more activation records included in the middleware computer program; identifying a program statement in the middleware computer program where a selected variable is last used; and inserting a nullification statement after the identified program statement, the nullification statement adapted to nullify the selected variable. The Office Action asserts that Sayag discloses the limitation of reading variables included in activation records, but correctly notes that

Sayag does not teach or suggest Applicant's "identifying" and "inserting" limitations (see Office Action, page 3). The Office Action then cites Kundu as disclosing identifying a program statement in a computer program where a variable is last used, and inserting a nullification statement after the identified program statement (see Office Action, page 4). Applicant respectfully disagrees. As discussed above, Kundu discloses "[t]techniques for making light-weight checkpoints in logs of streams of transactions" (see Kundu, Abstract). Kundu uses the redo logs that are typically used in database systems to log transactions, and purports to increase the usefulness of these redo logs for purposes such as data mining and replication of transactions. As discussed in detail in Kundu, a logical redo log is made from a physical redo log. Light-weight checkpoints are made and used in the logical redo log, and these light-weight checkpoints are used in data mining and replication (Kundu, paragraph [0041]).

The Office Action cites Kundu at paragraph [0051] as teaching the "identifying" step of Applicant's independent claims. The cited section of Kundu discusses "the information needed to make a logical redo log from a physical redo log" (Kundu, paragraph [0050]). This information includes various fields, such as session#, client#, server#, session_name, etc. The Office Action asserts that Kundu's "checkpoint_scn" is somehow equivalent to "identifying a program statement in the middleware computer program where a selected variable is last used." Applicant respectfully disagrees. Kundu's checkpoint_scn is the system change number, SCN, of the most recently-made

checkpoint in the physical redo log. As known to those skilled in the art, a checkpoint is a saved state of a program and its data, which can be used to restart the program at the point at which the checkpoint occurred. As explained in Kundu, "locations in physical redo logs are identified by system change numbers" (Kundu, paragraph [0051]). These system change numbers are used to identify locations within a physical redo log. Applicant fails to see how a system change number can be said to read on Applicant's step of "identifying a program statement in the middleware computer program where a selected variable is last used." Kundu is not at all concerned with determining where, within a program, a variable is last used. The physical redo logs in Kundu are concerned with taking a snapshot of a database program at a particular point in time. The variables used within that time frame may or may not continue to be used after the checkpoint is taken. Kundu simply is not concerned with determining where a variable is last used.

The cited section of Kundu at paragraph [0066] discusses producing a logical redo log. Checkpoints are made in the logical redo log by inserting checkpoint logical change records, LCRs, at the proper locations in the logical redo log. The Office Action asserts that inserting a logical change record into a logical redo log is equivalent to "inserting a nullification statement after the identified program statement," but then admits that Kundu does not disclose "the nullification statement adapted to nullify the selected variable." Applicant respectfully reminds the Examiner that MPEP § 2143.03 states that all the limitations of a claim must be considered. As stated in MPEP § 2143.03:

2143.03 All Claim Limitations Must Be Taught or SuggestedTo establish prima facie obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. In re Royka, 490 F.2d 981,180 USPQ 580 (CCPA 1974). "All words in a claim must be considered in judging the patentability of that claim against the prior art." In re Wilson, 424 F.2d 1382, 1385, 165 USPQ 494,496 (CCPA 1970). If an independent claim is nonobvious under 35 U.S.C. 103, then any claim depending therefrom is nonobviousness. In re Fine, 837 F.2d 1071,5 USPQ2d 1596 (Fed. Cir. 1988). Applicant clearly claims that the nullification statement that is inserted after the identified program statement is meant, "to nullify the selected variable." Kundu does not teach or suggest a nullification statement that nullifies a variable. Kundu discusses inserting logical change records into a logical redo log. A logical change record is not the same as a nullification statement. Even assuming, for the sake of argument (and Applicant does not agree that this is the case) that a logical change record could somehow be said to be equivalent to a nullification statement, the logical change records disclosed by Kundo do not nullify a variable. Kundu does not teach or suggest nullifying a variable, and Kundu certainly does not teach or suggest nullifying a variable by inserting a nullification statement after a program statement in which the variable is last used. Further, the Office Action admits that Kundu's logical change records do not nullify variables (see Office Action, page 5). Applicant specifically teaches and claims a nullification statement that nullifies a selected variable, and yet the Office Action admits that Kundu does not teach this aspect of Applicant's claim.

The Office Action then uses yet another reference, i.e. Kolawa, to disclose, "the nullification statement adapted to nullify the selected variable." However, the cited section of Kolawa at col. 5, lines 40-61, discusses a routine for performing leak searching. Allocated memory space is tracked using reference counting. A reference count is assigned to a pointer, and if the reference count becomes zero, it indicates that a memory leak may exist. Setting a reference count to zero does not nullify any variables, nor does it nullify the pointer. As clearly shown in Figure 7 of Kolawa, when the reference count equals zero, it indicates that there may be a memory leak, and thus it is time to do a memory search in order to determine if there is a memory leak. Using a reference count to determine that a potential memory leak exists is simply not the same as "inserting a nullification statement after the identified program statement, the nullification statement adapted to nullify the selected variable," as taught and claimed by Applicant. Kolawa simply has nothing to do with nullifying variables.

Because none of the cited references, either alone or in combination, teach or suggest all the elements of independent claim 1, Applicant respectfully submits that independent claim 1 is patentable over the cited references. Independent claims 10, 15, and 24-26 include limitations similar to those in independent claim 1, and are therefore patentable for at least the reasons discussed above with regard to claim 1. Therefore, Applicants respectfully submits that independent claims 1, 10, 15, and 24-26, and the claims which depend from them, are patentable over Sayag in view of Kundu and Kolawa.

Notwithstanding the patentability of independent claims 24-26 based on the above discussion, Applicant would like to discuss these claims in further detail. Using independent claim 24 as an exemplary claim, Applicant claims the following: reading one or more variables included in one or more activation records included in the middleware computer program; identifying a program statement in the middleware computer program where a selected variable is last used; inserting a nullification statement after the identified program statement, the nullification statement adapted to nullify the selected variable; writing a plurality of program statements, including the identified program statement, to a resulting code file; and writing the nullification statement to the resulting code file in a position subsequent to the identified program statement.

As discussed above, the prior art does not teach or suggest the "identifying" and "inserting" steps of Applicant's independent claims. Applicant further submits that none of the prior art, either alone or in combination, teaches or suggests "writing the nullification statement to the resulting code file in a position subsequent to the identified program statement," as taught and claimed by Applicants. The Office Action cites various sections of Sayag as disclosing this element. However, the Office Action also admits that Sayag does not disclose "identifying a program statement in the middleware computer program where a selected variable is last used," and "inserting a nullification statement after the identified program statement, the nullification statement adapted to nullify the selected variable" (see Office Action, page 3). Applicant is at a loss to understand how, if Sayag does not teach identifying a program statement where a

selected variable is last used or inserting a nullification statement after the identified program statement, Sayag can then be said to teach writing this nullification statement to a resulting code file in a position subsequent to the identified program statement. The cited sections of Sayag may discuss writing program statements to a file, but they do not discuss writing a nullification statement in a position subsequent to an identified program statement, where the identified program statement is specifically claimed to be the program statement in which a variable is last used, as taught and claimed by Applicant.

Independent claims 25 and 26 include limitations similar to those found in independent claim 24. For the reasons set forth above, Applicant respectfully submits that independent claims 24-26 are patentable over Sayag in view of Kundu and Kolawa.

Examiner's Response:

(3) In response to applicant arguments that the cited references do not teach or suggest all the limitations of applicant claims, Examiner states and is aware that: "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). >"When a claim covers several structures or compositions, either generically or as alternatives, the claim is deemed anticipated if any of the structures or compositions within the scope of the claim is known in the prior art." *Brown v. 3M*, 265 F.3d 1349, 1351, 60 USPQ2d 1375, 1376 (Fed. Cir. 2001) (claim to a system for setting a computer

clock to an offset time to address the Year 2000 (Y2K) problem, applicable to records with year date data in "at least one of two-digit, three-digit, or four-digit" representations, was held anticipated by a system that offsets year dates in only two-digit formats). See also MPEP § **2131.02**.< "The identical invention must be shown in as complete detail as is contained in the ... claim." *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). ***The elements must be arranged as required by the claim, but this is not an ipsissimis verbis test, i.e., identity of terminology is not required***. *In re Bond*, 910 F.2d 831, 15 USPQ2d 1566 (Fed. Cir. 1990).

(4) In response to applicant's statement and argument: "2143.03 All Claim Limitations Must Be Taught or SuggestedTo establish prima facie obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. In re Royka, 490 F.2d 981,180 USPQ 580 (CCPA 1974). "All words in a claim must be considered in judging the patentability of that claim against the prior art." In re Wilson, 424 F.2d 1382, 1385, 165 USPQ 494,496 (CCPA 1970). If an independent claim is nonobvious under 35 U.S.C. 103, then any claim depending therefrom is nonobviousness. In re Fine, 837 F.2d 1071,5 USPQ2d 1596 (Fed. Cir. 1988).

As stated above (see examiner response number 1) Examiner recognizes and have established prima facie of obviousness, in which Kundu teaches "identifying a program statement in the program where the variable is last used" paragraph [0051], wherein the remaining field contain information about the current state of the session, wherein

locations in physical redo logs are identified by system change numbers, or SCN's. start_scn and end_scn are the SCN's of the physical redo blocks at which the logical redo log made for the session will begin and end and checkpoint_scn is the SCN of the most recently-made checkpoint in the logical redo log, which is equivalent to identifying a program statement in the program where the variable is last used, and wherein a checkpoint is interpreted to be an identified snapshot of a database or a point at which the transactions against the database have been frozen, Kundu); and Kundo teaches "inserting a nullification statement after the identified program" (paragraph [0066], wherein the builder also makes checkpoints in logical redo log by inserting checkpoint logical change records, i.e. LCR at the proper locations in logical redo log, and wherein a redo log is interpreted to be a set of files that record all changes made to an database and a checkpoint is interpreted to be an identified snapshot of a database or a point at which the transactions against the database have been frozen, which is equivalent to nullifying the identified last used variable, Kundu). Therefore, It would have been obvious to one of the ordinary skill in the art at the time of the invention was made to incorporate Kundu {see abstract} teaching into Sayag system to recognize when a variable was last used within a program. A skilled artisan would have been motivated to do so for allocating more space within memory to store data as well as maintaining memory management in a system.

AND wherein: Sayag in view of Kundu do not teach, "the nullification statement adapted to nullify the identified last-used variable".

On the other hand, Kolawa disclose the nullification statement adapted to nullify the identified last-used variable (column 5, lines 40-61, wherein track using allocated memory space using reference counting, if value is stored in a pointer and so forth, and wherein if the reference count for the pointer variable becomes zero, wherein zero is equivalent to nullify, Kolawa). Therefore, It would have been obvious to one of the ordinary skill in the art at the time of the invention was made to incorporate Sayag in view of Kundu teachings into Kolawa system. A skilled artisan would have been motivated to combine as suggested by Kolawa at {column 1, lines 60-62 and column 2, lines 6-9, Kolawa}, for providing a code to identify and track memory space. As a result, for detecting leak memory space in a computer program.

Prior Art of Record

(The prior art made of record and not relied upon is considered pertinent to applicant's disclosure)

1. Sayag	(US Patent No. 6,898,602)
2. Kundu et al.	(US Publication No. 2004/0210577)
3. Kolawa et al	(US Patent No. 5,842,019)

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Point of Contact

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Helene R. Rose whose telephone number is (571) 272-0749. The examiner can normally be reached on 8:00am - 4:30pm M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Don Wong can be reached on (571) 272-1834. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

HRR
Technology Center 2100
May 26, 2007

ALFORD KINDRED
PRIMARY EXAMINER